Massachusetts Institute of Technology
Department of Electrical Engineering and Computer Science

6.341: Discrete-Time Signal Processing

OpenCourseWare 2006

Lecture 7
IIR, FIR Filter Structures

Reading: Sections 6.1 - 6.5 in Oppenheim, Schafer & Buck (OSB).

Signal Flow Graphs

A linear time-invariant discrete-time system is in general represented by a linear constant-coefficient difference equation characterizing the input-output relation of the system. As a network structure, such a difference equation can be represented by a block diagram or a signal flow graph. A signal flow graph is a network of directed branches that connect at nodes. It is equivalent to block diagrams which we are already familiar with, except for a few notational differences. As examples, OSB Figures 6.8 and 6.9 depict the general form of signal flow graphs.

In a signal flow graph, the value carried by a specific branch is equal to the value of its originating node. Nodes in signal flow graphs represent variables. The value carried by a specific node is the sum of all branches coming into it. If there is only one entering branch, the node is a "branching note" rather than a "summing node." Two special types of nodes exist: source nodes have no entering branches, they present external signal sources; sink nodes have only entering branches, they extract output from a graph. These are both labelled accordingly in OSB Figure 6.11, which is the signal glow graph corresponding to the first order system in OSB Figure 6.10. By convention, the delay element has been represented by a branch gain of z^{-1}.

The signal flow graph representation of a LTI system is not unique. In fact, for any given rational system function, equivalent sets of difference equations and network structures exist. In practical implementations, factors such as number of multipliers and adders, regularity of the structure, and finite-word-length effects are taken into account when deciding which network structure to use.

IIR Filter Structures

Direct Forms (I & II)

Consider the following general form of a difference equation and the corresponding system transfer function:

$$y[n] = \sum_{k=0}^{M} b_k x[n-k] + \sum_{k=1}^{N} a_k y[n-k] \quad \Rightarrow \quad H(z) = \frac{\sum_{k=0}^{M} b_k z^{-1}}{1 - \sum_{k=1}^{N} a_k z^{-1}} \, .$$

If we draw the signal flow graph of this Nth-order system by cascading a feedforward section and a feedback section, we obtain its *Direct Form I* structure shown in OSB Figure 6.14. Note that the feedforward section determines the zeros of the transfer function, while the feedback section gives the poles. Interchanging the order of the feedforward and feedback sections and combining the delay elements give the *Direct Form II* structure shown in OSB Figure 6.15.

Since delay elements correspond to physical memories in actual implementation, direct form II structures require less state memory than the direct form I implementation. However, the total memory requirement for both forms are similar, because direct form II structures need more cache memory during computations.

Transposed Forms

Using signal flow graphs, we can transform a given system into a different network structure while maintaining the same system function. One such transformation is transposition.

Transposition Theorem

1. Reverse direction of all branches
2. Interchange input and output

For single-input single-out systems, interchanging the input and output nodes after reversing the flow graph gives the same transfer function as the original system. OSB Examples 6.7 and 6.8, and OSB Figures 6.24-6.30 illustrate transposition of basic systems. Although the transfer functions remain the same, different network structures represent different algorithms, which are equivalent under ideal infinite precision arithmetic. With finite precision arithmetic, the implementation structure determines internally generated noise which affects the overall system behavior.

Cascade Form

Rather than deriving the signal flow graph directly from the system function as in the direct form cases, we could also factor the denominator and numerator of the system transfer function into first-order and second-order subsystems:

$$H(z) = \prod_{k=1}^{N_s} \frac{b_{0k} + b_{1k}z^{-1} + b_{2k}z^{-2}}{1 - a_{1k}z^{-1} - a_{2k}z^{-2}} \,.$$

OSB Figure 6.18 is an example of the resulting cascade structure. This is a sixth-order system with direct form II realization for each of its second-order subsystems.

Parallel Form

Equivalently, expressing the transfer function as a sum using partial fraction expansion gives a parallel structure:

$$H(z) = \sum_{k=0}^{N_p} C_k z^{-k} + \sum_{k=1}^{N_s} \frac{e_{0k} + e_{1k} z^{-1}}{1 - a_{1k} z^{-1} - a_{2k} z^{-2}} .$$

OSB Figure 6.20 shows a parallel form structure for a sixth-order system. See Section 6.3 for a more detailed explanation.

FIR Filter Structures

FIR systems are special cases of IIR systems, which can be structured into direct, cascade, or parallel forms. There also exist additional forms specific to FIR systems.

Direct Form

A causal FIR system described by the following difference equation has all of its poles at the the origin:

$$y[n] = \sum_{k=0}^{M} b_k x[n-k] .$$

As such, its direct form I & II implementations reduce to the flow graph shown in OSB Figure 6.31. Such a structure is usually called a *tap-delay line filter* structure or a *transversal filter* structure. Furthermore, applying the transposition theorem gives the equivalent system in OSB Figure 6.32. Note that these are non-recursive, ie. there is no feedback. The implementation of FIR systems, nonetheless, is not necessarily always nonrecursive, since pole-zero cancellation may exist.

Direct Form Structure for Linear-Phase FIR Systems

In previous lectures we have shown that a causal generalized linear phase FIR system satisfies the following symmetry (anti-symmetry) condition, depending on the type of the system:

$$h[M-n] = h[n] \quad \text{or} \quad h[M-n] = -h[n] \qquad n = 0, 1, \ldots, M .$$

Using this special property, can we further simplify the tap-delay line filter structure to reduce the number of multipliers required? The answer is yes. Consider a type I system where the transfer function $H(z)$ is of even degree and symmetric. Rewrite $H(z)$ as:

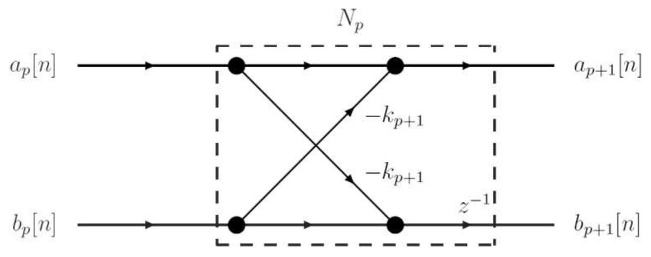
$$\begin{aligned} H(z) &= \sum_{n=0}^{M} h[n] z^{-n} = h[0](1 + z^{-M}) + h[1](z^{-1} + z^{-(M-1)}) + \ldots \\ &= \sum_{n=0}^{M/2-1} h[n](z^{-n} + z^{-(M-n)}) + h[M/2] z^{-M/2} . \end{aligned}$$

This equation suggests the realization shown in OSB Figure 6.34. Similarly, structures can be derived for type II, III and IV systems. See OSB Figure 6.35 for an example of a type III system where the order M is odd.

Lattice Filters

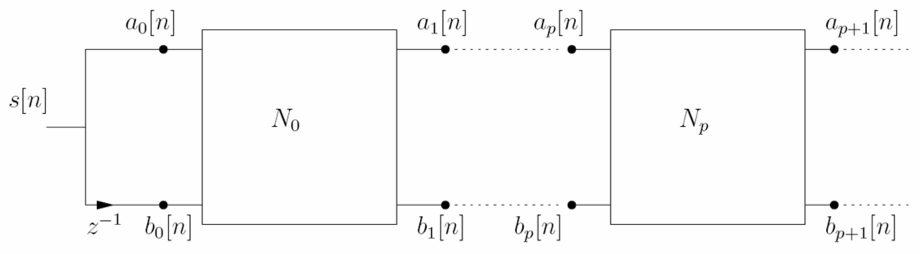
We have previously shown that a single-input single-output (SISO) system can be implemented by cascading SISO direct-form structured subsystems. In this section, we show another possible implementation, called lattice filters, where the subsystems are two-port LTI systems of particular forms. The overall cascade is converted to the required SISO form by terminating both ends according to filter type specific rules.

All-zero (FIR) Lattice Filters the basic two-port section in an FIR lattice filter has the following non-recursive butterfly signal flow graph structure:



One section of lattice structure for FIR lattice filters

For the overall system, the input is fed into the two input ports of the first stage, while the output is taken as that of the top branch of the last stage:



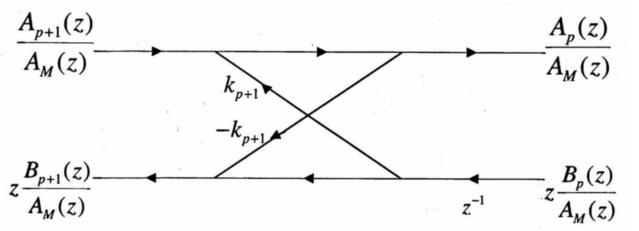
General form of a lattice filter

The coefficients $k_1, k_2, \ldots, k_{p+1}$ are called the k-parameters of the lattice structure. If the input $s[n]$ is a unit impulse, it can be shown that the intermediate transfer functions in this FIR lattice filter satisfy the following recursive equations:

$$A_0(z) = 1 \qquad A_{p+1}(z) = A_p(z) - k_{p+1}B_p(Z)$$
$$B_0(z) = z^{-1} \quad B_{p+1}(z) = z^{-1}[B_p(z) - k_{p+1}A_p(z)] \; .$$

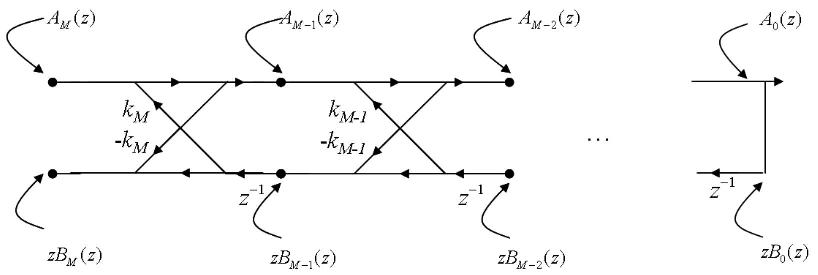
Algorithms exist for analyzing an FIR lattice filter to obtain its transfer function, or constructing an FIR lattice structure (ie. calculating the k-parameters) from a given rational system function. These will be studied later in this course.

All-pole Lattice Filters Given the structure of an all-zero lattice filter as discussed in the previous section, by successively reversing each lattice section, we can obtain an all-pole lattice filter. The signal flow graph for a reversed lattice section is



One section of lattice structure for all-pole filters

We have scaled all signals by $A_M(z)$ because assuming the input is at the $p+1 = M$-th stage and the output is at $p = 0$, setting $A_{p+1}(z) = A_M(z)$, and $A_0(z) = 1$ gives the desired all-pole filter: $\frac{A_0(z)}{A_M(z)}$. The following figure shows the overall structure of an all-pole lattice filter. Note since $B_0(z) = z^{-1}$, the bottom branch of the final lattice section should be connected to the top branch.



All-pole lattice filter

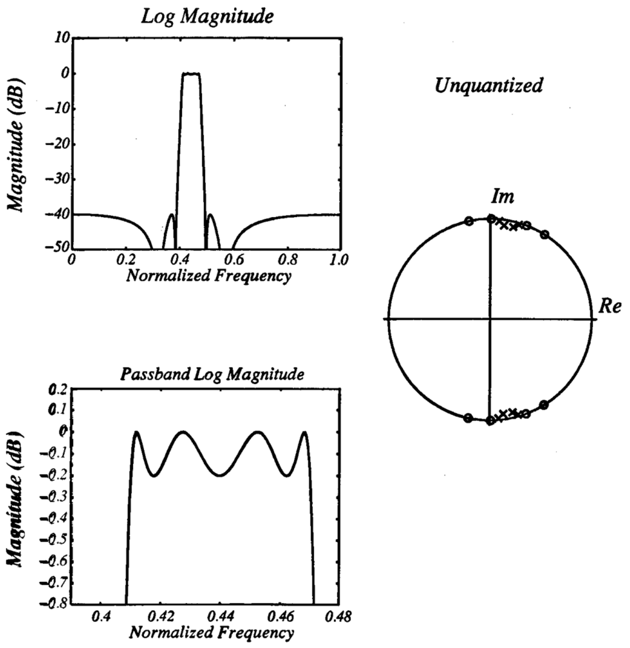
An important property of all-pole lattice filters is that the systems are stable if and only if all of the k-parameters have magnitudes less than 1, ie. $|k_i| < 1$ for all i.

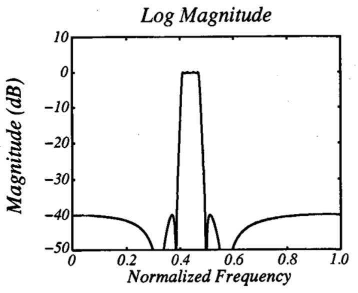
Effects of Coefficient Quantization

So far, we have looked at different ways of implementing the same difference equation. Although theoretically equivalent, each implementation may behave differently in the presence of finite precision arithmetic. Numerical problems can be introduced by filter coefficient quantization and signal quantization.

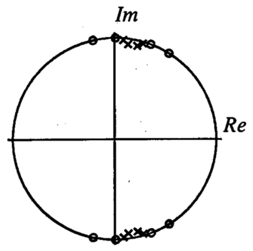
As an illustration of the coefficient quantization effect, OSB Figure 6.40 displays the frequency responses of three different implementations for a 12th-order bandpass IIR elliptic filter with unquantized and quantized coefficients. Here 32-bit floating-point accuracy has been defined as "unquantized," and the 16-bit quantized coefficients have been listed in OSB Tables 6.1 and 6.2. See OSB Section 6.7.2 for detailed analysis of this example.

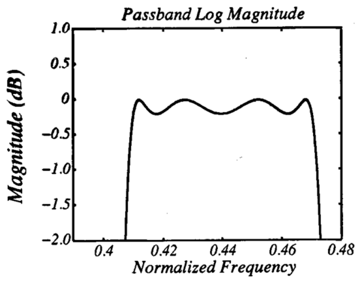
Filter coefficient quantization causes the poles and zeros of the system to shift, consequently distorting its frequency response. The next set of figures compares the log magnitudes and pole-zero plots of an 8th-order bandpass filter under different implementation forms and various quantization accuracies. Note when the coefficients are quantized to 12 or 10-bits in the direct form implementation, some poles move to the outside of the unit circle, making the overall system unstable.

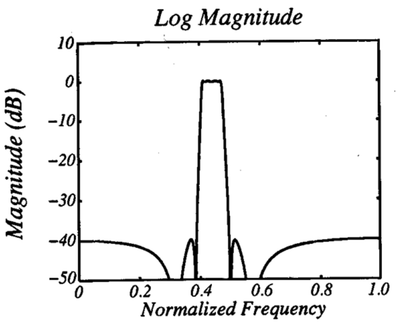




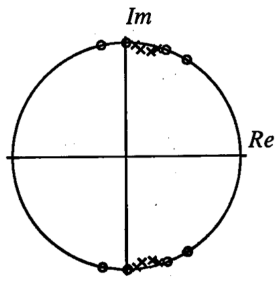
Cascade Form
16 bit quantization

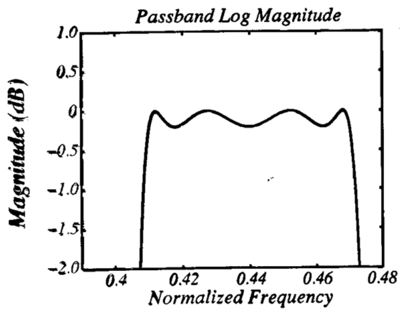


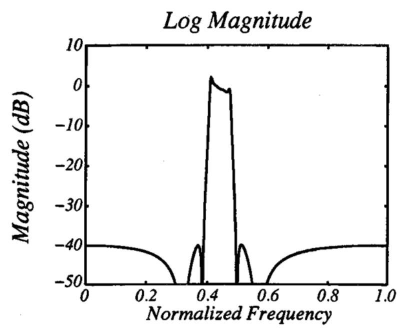




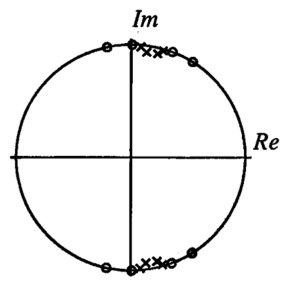
Parallel Form
16 bit quantization

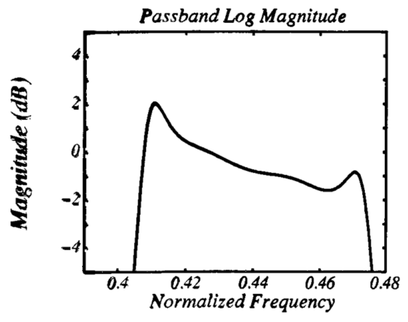


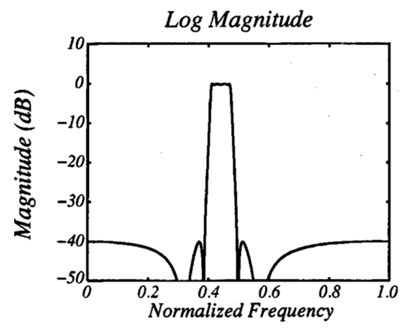




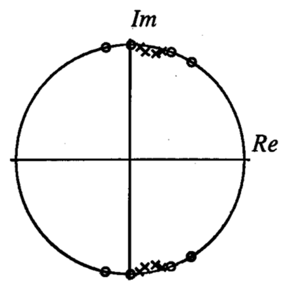
Direct Form
16 bit quantization

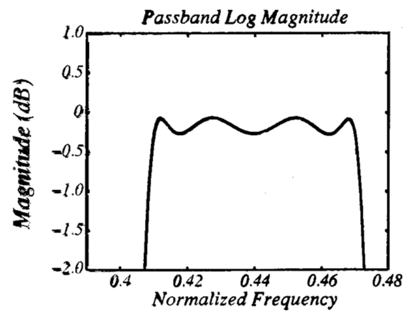


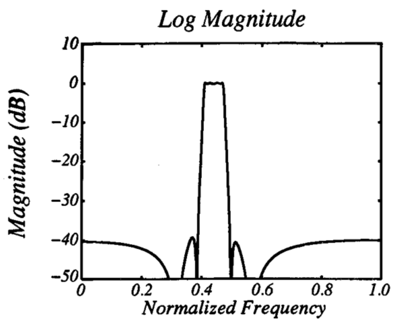




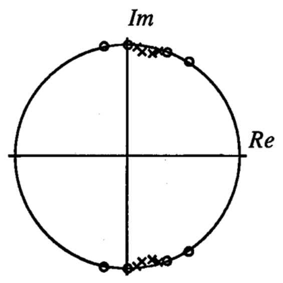
Cascade Form
14 bit quantization

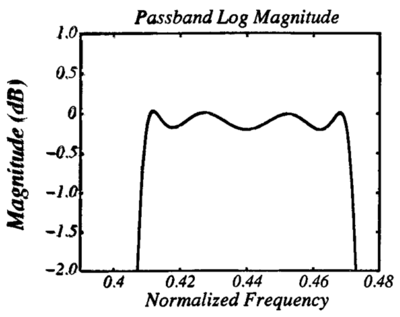


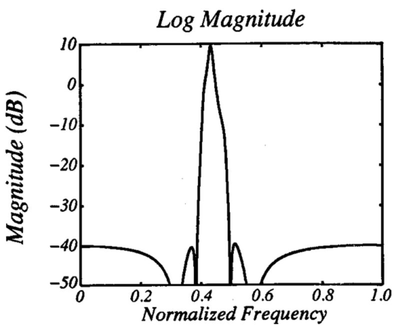




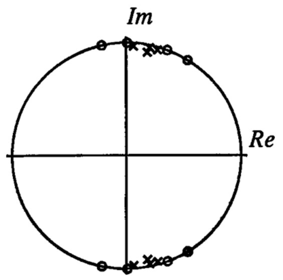
Parallel Form
14 bit quantization

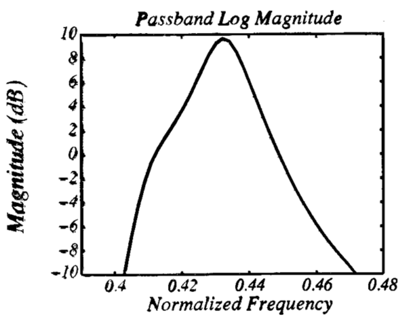


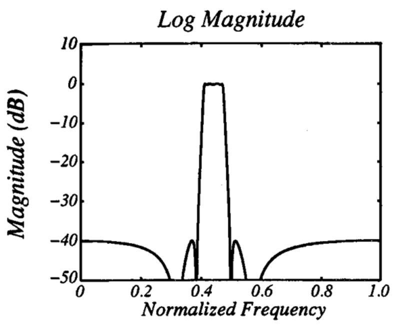




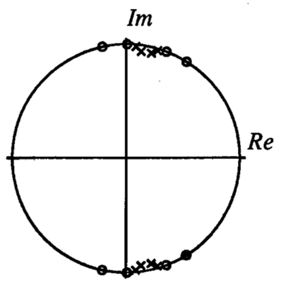
Direct Form
14 bit quantization

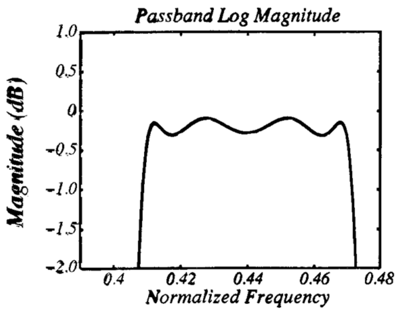


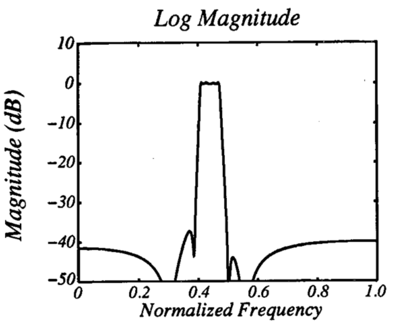




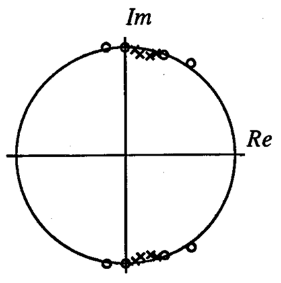
Cascade Form
12 bit quantization

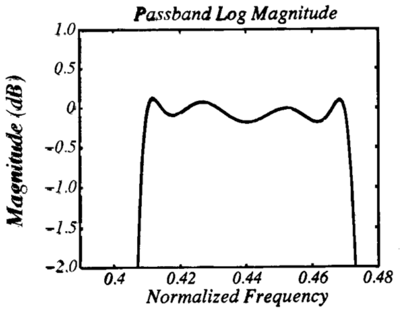


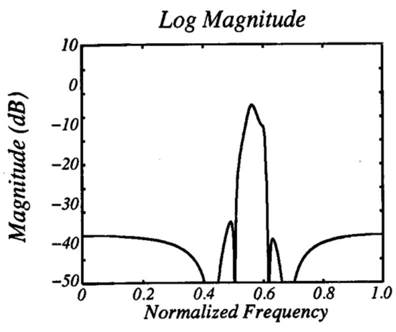




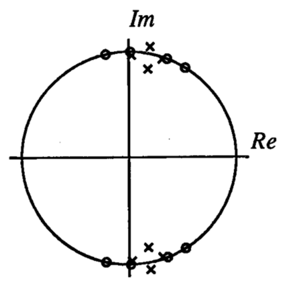
Parallel Form
12 bit quantization

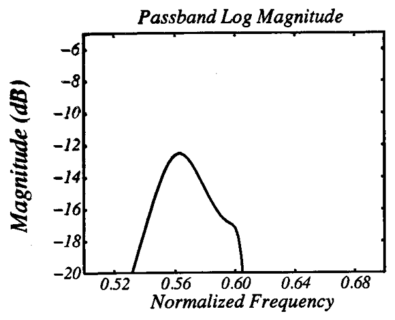


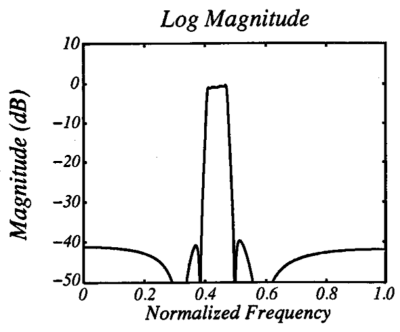




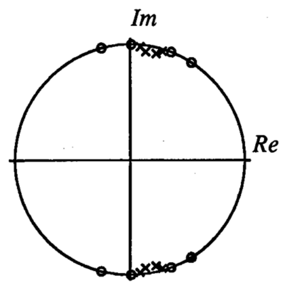
Direct Form
12 bit quantization

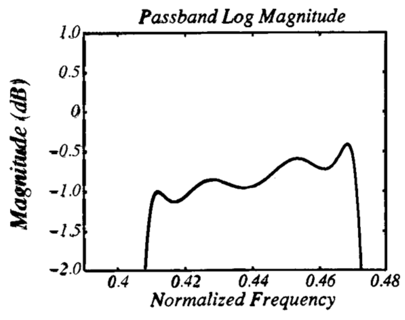


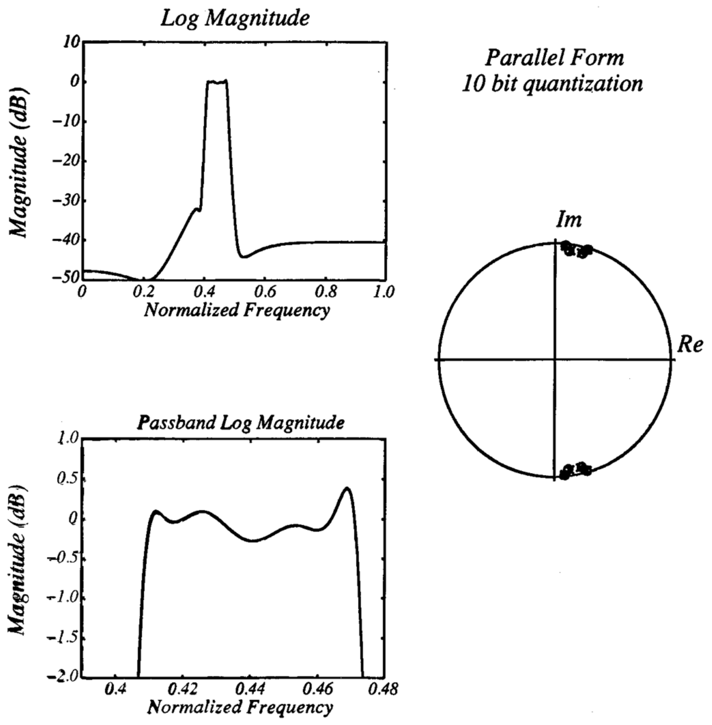




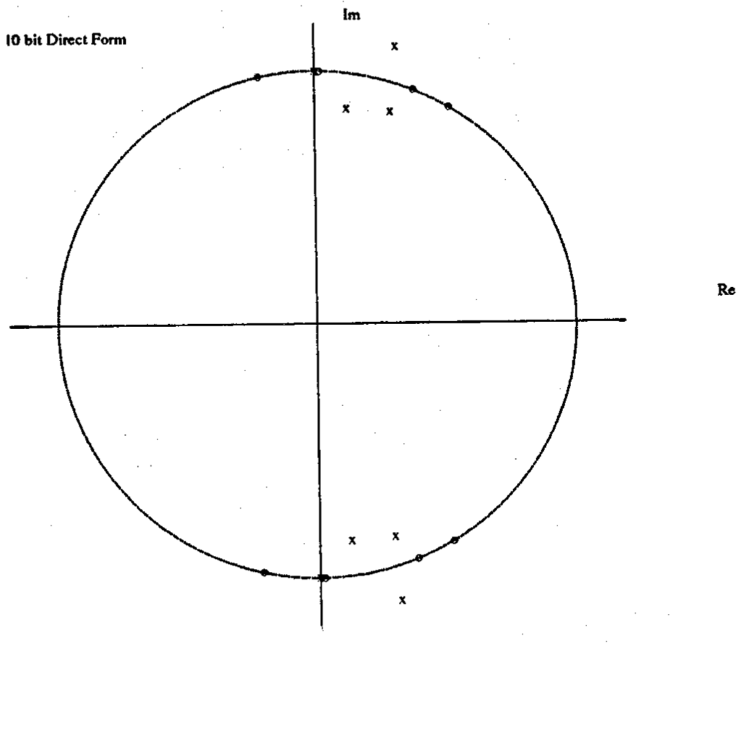
Cascade Form
10 bit quantization





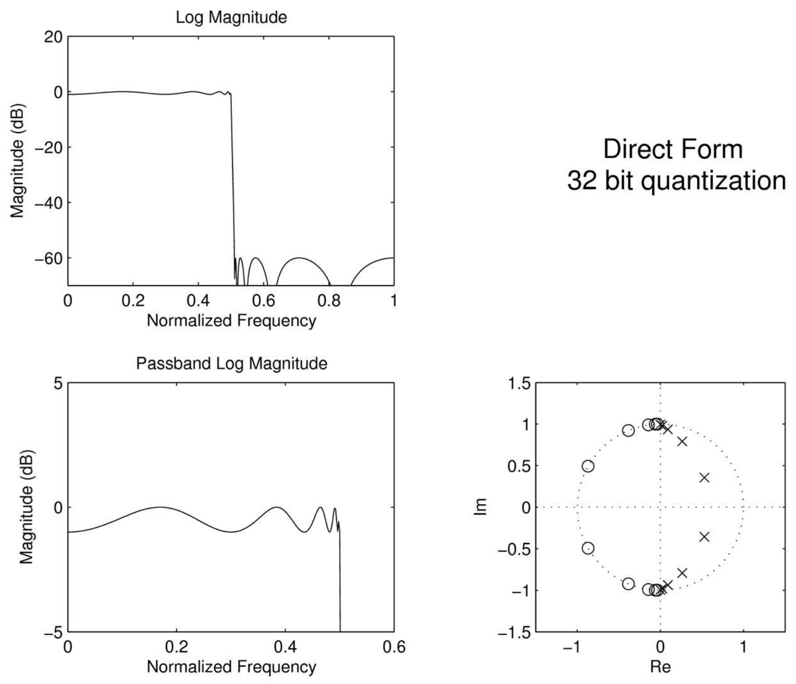


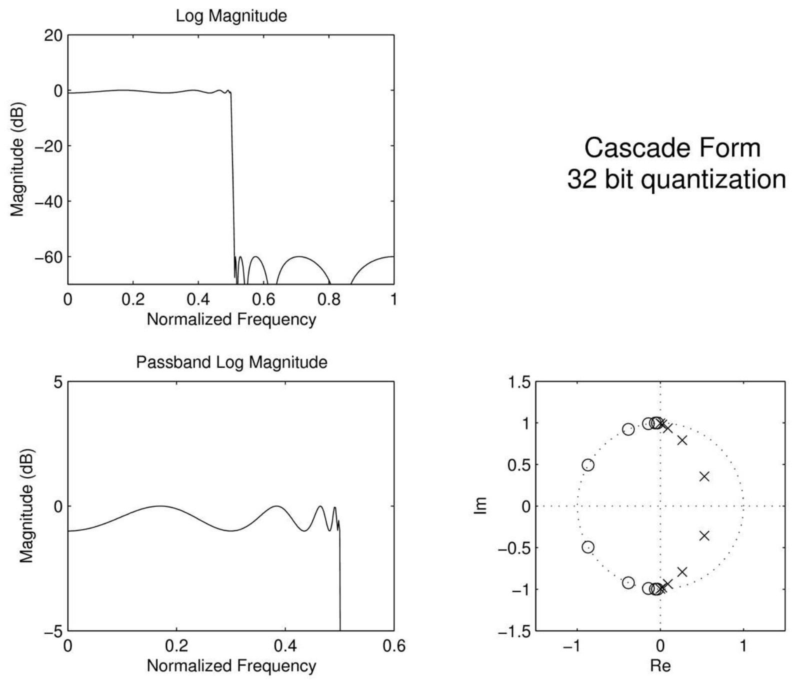
Log Magnitude
Magnitude (dB)
10
0
-10
-20
-30
-40
-50
0
0.2
0.4
0.6
0.8
1.0
Normalized Frequency
Parallel Form
10 bit quantization
Im
Re
Passband Log Magnitude
Magnitude (dB)
1.0
0.5
0
-0.5
-1.0
-1.5
-2.0
0.4
0.42
0.44
0.46
0.48
Normalized Frequency

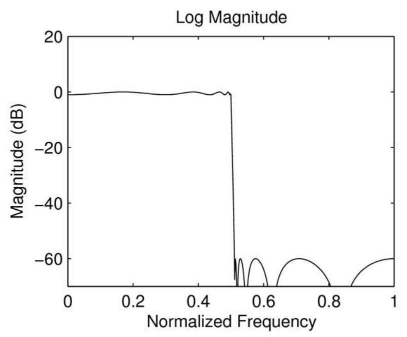


10 bit Direct Form
Im
Re

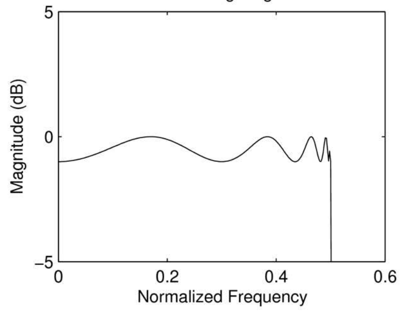
The next example is an 8th-order lowpass filter:

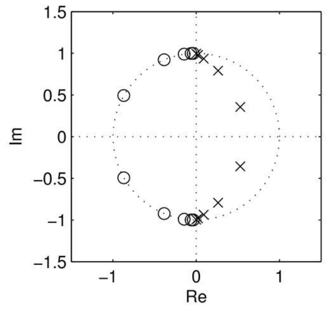


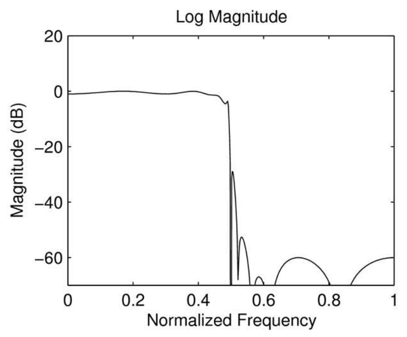




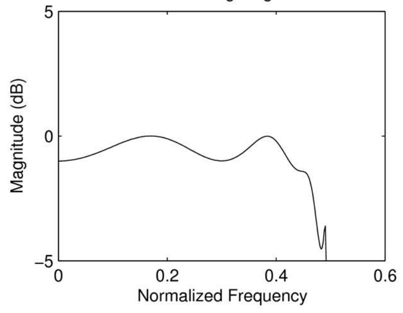
Parallel Form
32 bit quantization

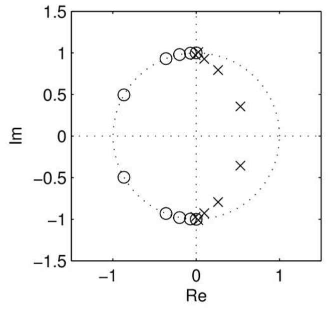


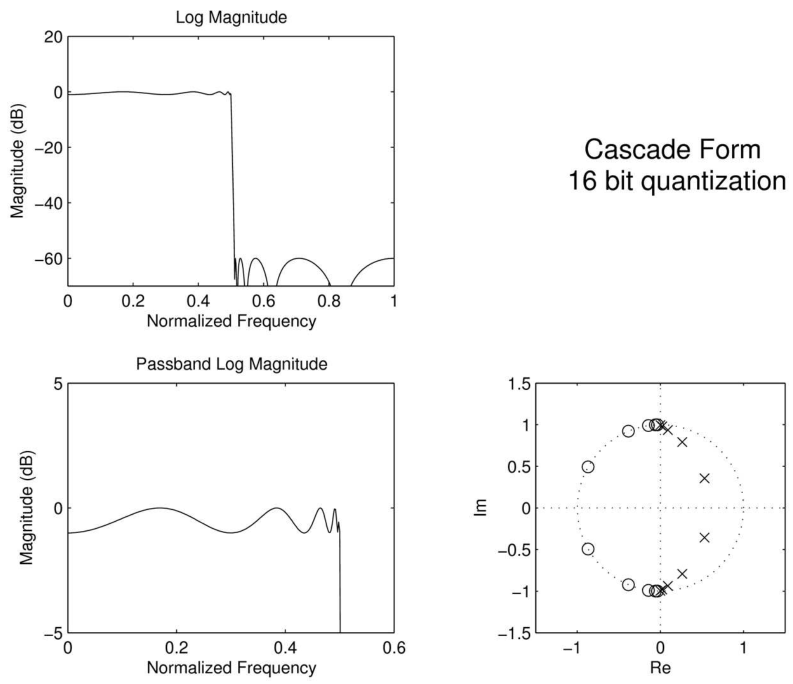


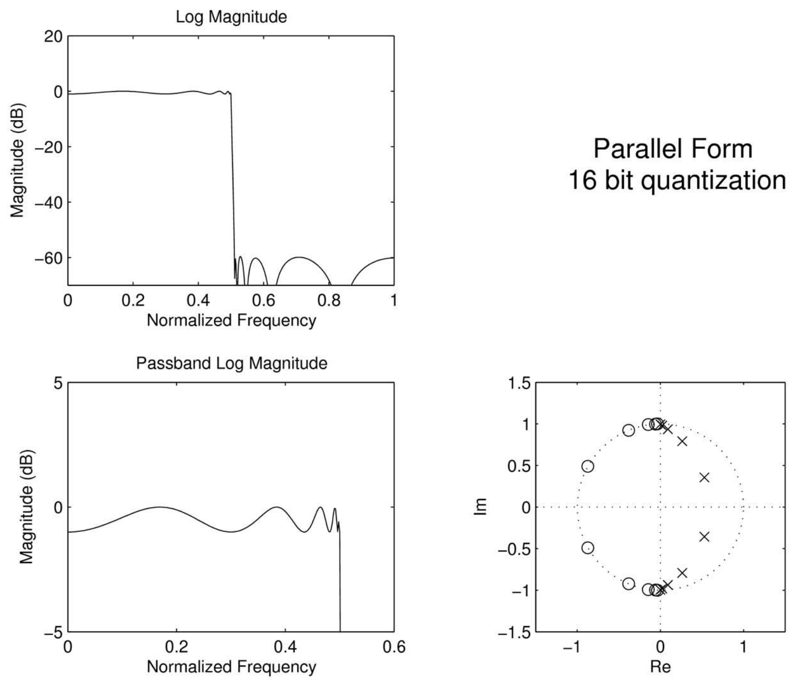


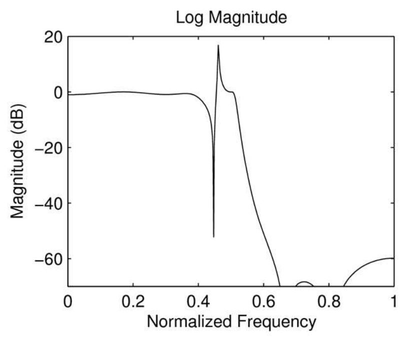
Direct Form
16 bit quantization



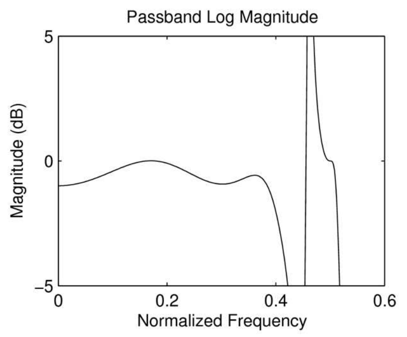


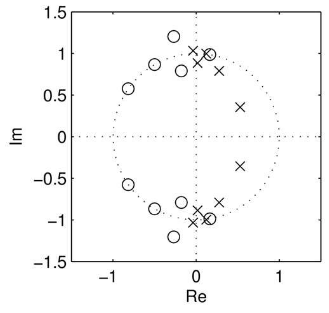

Log Magnitude
Magnitude (dB)
Normalized Frequency
Cascade Form
16 bit quantization
Passband Log Magnitude
Magnitude (dB)
Normalized Frequency
Im
Re

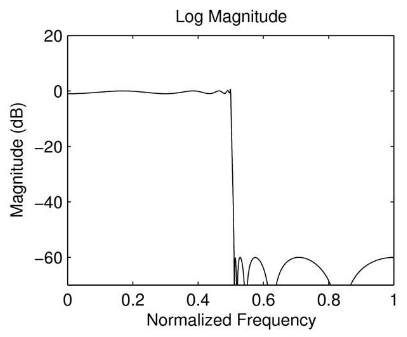

Log Magnitude
Magnitude (dB)
Normalized Frequency
Parallel Form
16 bit quantization
Passband Log Magnitude
Magnitude (dB)
Normalized Frequency
Im
Re



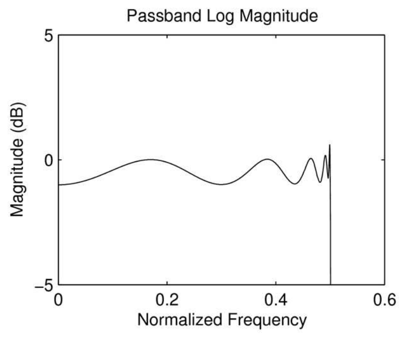
Direct Form
12 bit quantization

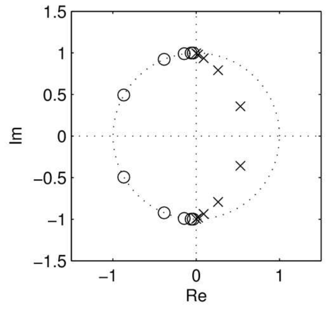


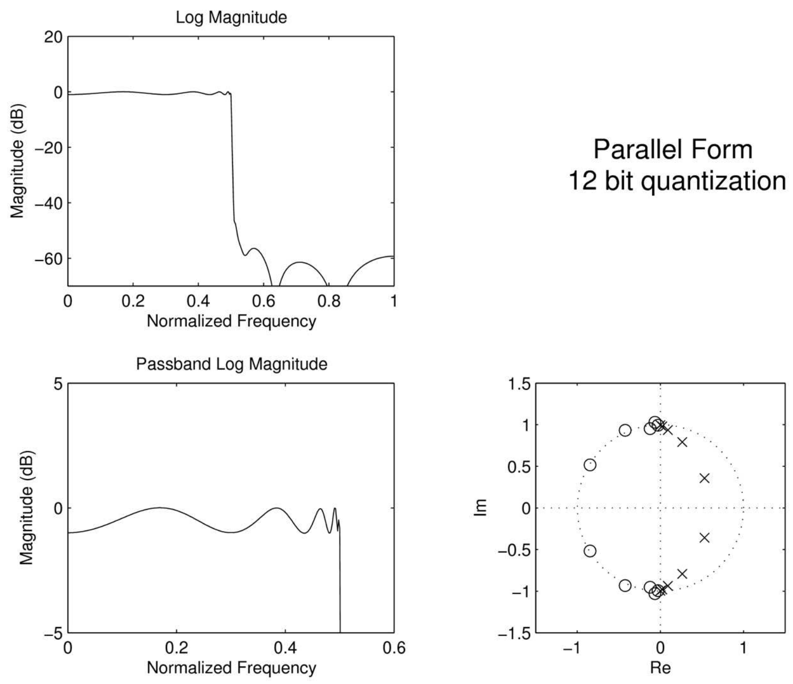




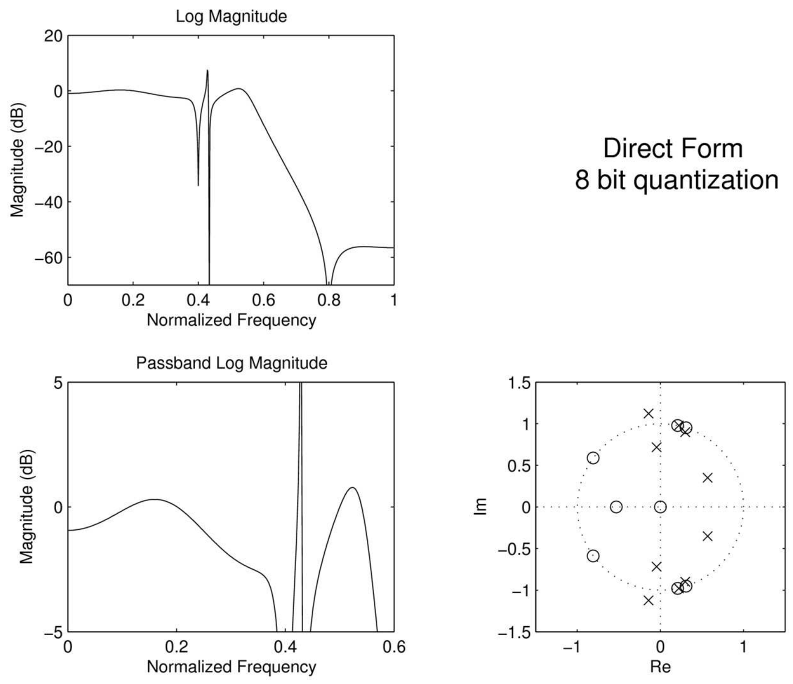
Cascade Form
12 bit quantization



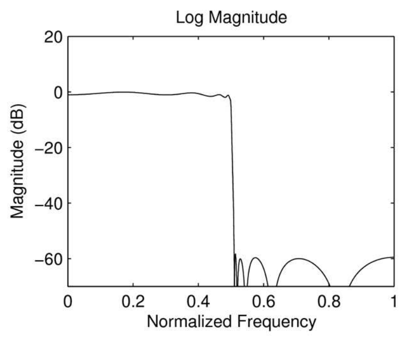




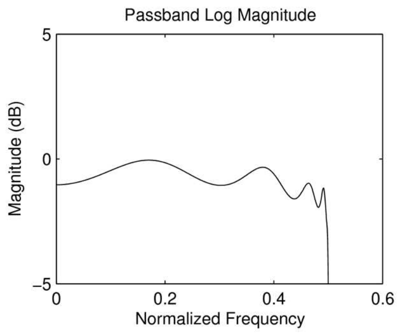
Parallel Form
12 bit quantization

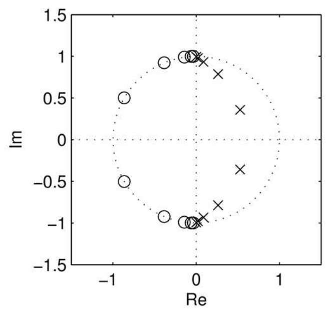


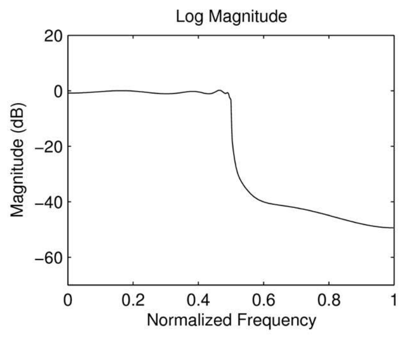
Direct Form
8 bit quantization



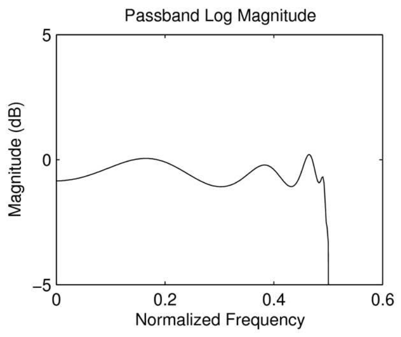
Cascade Form
8 bit quantization

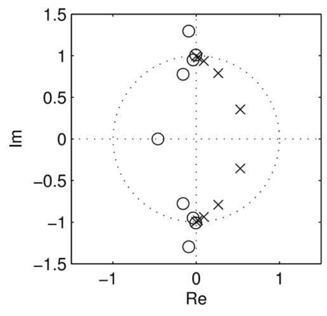


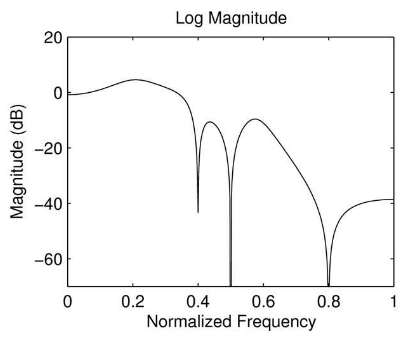




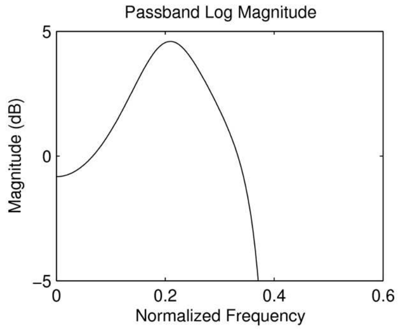
Parallel Form
8 bit quantization

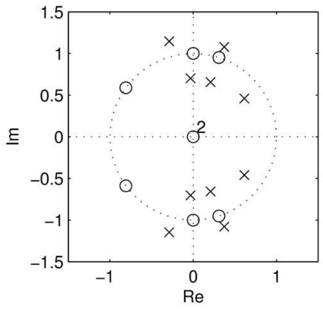


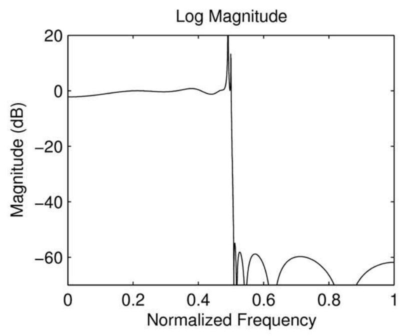




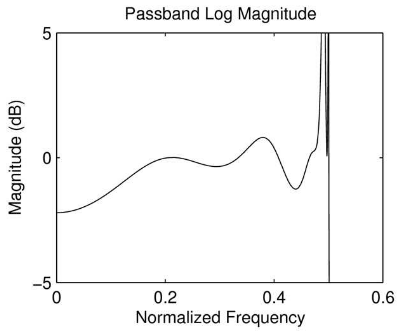
Direct Form
6 bit quantization

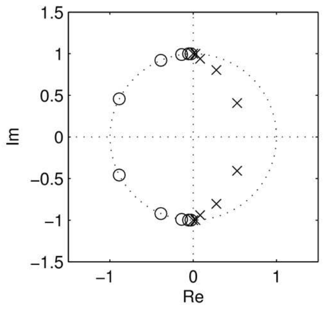




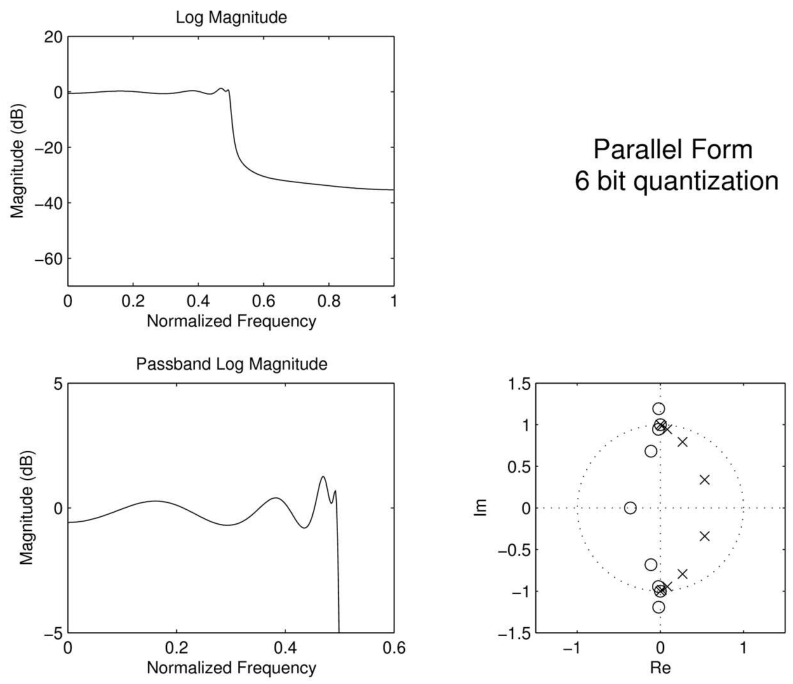


Cascade Form
6 bit quantization





Parallel Form
6 bit quantization


Log Magnitude
Magnitude (dB)
Normalized Frequency
Passband Log Magnitude
Magnitude (dB)
Normalized Frequency
Im
Re